REMARKS/ARGUMENTS

The application as filed contains no paragraph numbering. All references to enumerated paragraphs of the Specification correspond to numbering included in the published application.

Objections to the Drawings

Included herewith are replacement sheets 1/3, 2/3 and 3/3 containing FIGS. 1, 2, and 3, respectively. Also included herewith are annotated versions of the corresponding published figures. The amendments to FIGS. 1 and 2 add no new matter. FIGS. 1 and 2 are revised to better correspond with the text in Specification paragraphs [0085], [0086], [0088], [0091] through [0094], and [0096] through [0099] as currently amended. Replacement sheet 3/3 contains no amendments but is reproduced for added clarity.

The text box in FIG. 1 identified with number 3 now reads "Tracing Step." The text box in FIG. 1 identified with number 4 now reads "Filtering Step," and the text box in FIG. 1 identified with number 5 now reads "Analyzing Step." These terms concur with Specification paragraph [0027] and paragraph [0085] as amended to recite the executable software steps. One skilled in the art would understand that a computer software program comprises a series of executable steps. These amendments add no new matter.

The text box in FIG. 1 identified with number 11 now reads "Transformation Step." This amendment corresponds to Specification paragraph [0023] and paragraph [0085] as amended for consistency with at least Specification paragraph [0023].

The text box in FIG. 1 identified with number 31 now reads "Source Code (Program Language)" to better correspond with the text box associated with number 33 and to better correspond with the Specification. For example, support for this modification exists at least at paragraph [0094] of the specification, reproduced here in pertinent part for convenience:

> [0094] One embodiment of the invention may produce source code 31 that may mimic what the user did during the recording and tracing phase. The source code 31 may be used on a client machine or on a server machine.

The text boxes labeled 33 and 35 are respectively amended to read "Modified Source Code" and "Source code playback in Browser Plug-In" and now better correspond with the Specification, for example at paragraphs [0055], [0094], [0096] and [0099].

The examiner objects to the drawings as failing to comply with 37 CFR 1.84(p)(5) because they do not include reference element "25" listed on page 28 of the specification, within a paragraph now bearing label [0092] in the published application. This response includes a revised paragraph [0092] which deletes this stray reference element from the Specification. The drawings accordingly contain no reference element 25.

FIG. 1 now also comprises an element 34 representing playback of the source code in a dedicated WIN32 Application. Additionally, optional debug element 37 is show properly placed between the generation of source code (elements 31 and 33) and playback of the generated source code (elements 34 and 35). One skilled in the art would understand that debugging source code is an optional step in software development. Support for these amendments exist at least at Specification paragraphs [0076] and [0099], as amended, reproduced here in pertinent part for convenience:

> [0076] Still another advantage of the present invention is... ***to provide full interactive debugging capability to the software code that is generated during the tracing and code generation phases***. [Emphasis added]
>
> [0099] ~~Figure~~FIG. 3 illustrates the relationship of various software modules of the computer software program 1 in an embodiment. The software modules may be divided into those that run during the analysis data capture phase, and software that may run either on a client and/or on a server machine during playback of generated source code 31. ***Client playback may be either via a dedicated win32 application 34, or a web browser plug-in 35 and/or browser helper object.*** [Emphasis added]

Replacement sheet 2/3 more clearly corresponds to the method steps supporting the method of use of the computer software program of the present invention as described in the Specification. This revised FIG. 2 contains no new matter and support for these changes exists at least in Specification paragraphs [0088], [0076], [0089], [0092] and [0099].

Specification

None of the revised paragraphs of the Specification included in this amendment contain new matter. The following remarks address substantive amendments to the Specification by way of replacement paragraphs:

Page 2 of the instant office action contains the following instruction:

> The use of the trademark JAVA, JAVASCRIPT, VISUAL BASIC, and COLD FUSION has been noted in this application. It should be capitalized wherever it appears and be accompanied by the generic terminology.

Accordingly, this amendment contains revised paragraphs [0001], [0016], [0017], [0023], [0044], [0045], [0068], [0070], [0071], [0084], [0087], [0088], [0091], [0095], and [0099]. These revised paragraphs now appropriately identify trademarks via uppercase lettering, and these trademark terms are accompanied by generic terminology.

Revised paragraphs [0016], [0027], [0069], and [0085] are amended for proper capitalization of the term "Internet Protocol (IP)."

Revised paragraph [0016] also contains clarification of the process for tracing IP network events and collecting associated data. Support for this amendment exists at least at paragraphs [0016] and [0027] of the published Specification and on page 6 of related provisional application 60/423,648.

Revised paragraphs [0084], [0085], [0088], [0090] through [0092], [0094], [0096], [0097], [0099] and [0100] contain amendments that clarify references used to identify the computer software program and the software robot comprising the source code. These amendments also remove the undefined term "AFT" from the Specification. Page 1 of related provisional application number 60/423,648 introduced the term "AFT" to represent the computer software program. Amendments to paragraphs [0085] and [0096] now include the generic terminology rather than the specific naming convention.

Revised paragraph [0085] now describes the computer software program as comprising executable steps. Support for this amendment exists at least at paragraphs [0023] and [0027]. One of ordinary skill in the art would understand that software programs comprise a series of executable steps. This amendment adds not new matter.

Revised paragraph [0087] contains an amendment that corrects the verb "includes" to "include" for purposes of noun verb agreement.

Revised paragraph [0088] now comprises an introductory sentence better introducing FIG. 2. Additionally, changes include repositioning the last two sentences within the paragraph for better elucidation of the described elements.

Revised paragraph [0089] properly recites the executable step elements consistent with amendments to paragraph [0085], for example. Additionally, paragraph [0089] contains an amendment that refines "events" as IP nework events. Support for this amendment exists at least at Specification paragraph [0091].

Revised paragraph [0090] properly recites the analyzing step and tracing step. Revised paragraph [0090] also includes amendments that improve sentence structure to better elucidate described elements.

Revised paragraph [0091] contains language further describing the function of the software hook. Support for this functional language exists throughout the specification, and for example at paragraphs [0027] and [0088]. Revised paragraph [0091] also contains an amendment clarifying the term "web browser" as synonymous with a "website browser".

Revised paragraph [0092] now includes an amendment to better describe the term "IP packets" as "IP data packets." Support for this amendment exists at least at Specification paragraph [0091]. Additionally, "collage" is corrected to "collate," which is consistent with the preceding sentence.

Revised paragraph [0094] correctly identifies the term "XML extract data." Additionally, paragraph [0094] now references revised FIG. 1 which better reflects the optional embodiment of modifying source code by way of inclusion of a broken line in FIG. 1 connecting the source code 31 and the modified source code 33: "This optional embodiment is shown by way of a broken line in FIG. 1 between the souce code 31 and the modified source code 33."

Revised paragraph [0095] correctly describes the XSL output. Support for this amendment exists at least at paragraph [0094]. Additionally, one skilled in the art would recognize the operability of XSL.

Revised paragraph [0096] contains amendments directed toward grammatical and structural changes to the text. Additionally, the adverb "automatically" now qualifies the generation of source code. Support for this amendment exists throughout the specification, for example at paragraph [0029] as described below with regard to amendments to claims 1 and 12 that overcome at least the rejections under 35 USC 112, second paragraph. Additionally, the debug function is properly described as debugging the automatically generated source code rather than the browser. Support for this amendment exists at least at Specification paragraphs [0076] through [0079]. For example, Specification paragraph [0076] explicitly describes debugging the source code:

> [0076] Still another advantage of the present invention is to provide an automated software robot generator and a method for using the same wherein a web browser plugin is used ***to provide full interactive debugging capability to the software code that is generated during the tracing and code generation phases***. [Emphasis added]

Applicant respectfully submits that amendments to paragraph [0096] add no new matter.

Revised paragraphs [0097], [0098] and [0093] contain typographical corrections, replacing "clink" with "click."

Revised paragraph [0100] contains an amendment to sentence structure that adds no new matter.

Lastly, paragraphs [0085] through [0100] are modified for proper agreement and consistent use of numbered terms. Similar modifications are provided in FIGS. 1 and 2.

Rejections under 35 USC 101

Claims 1-11 are rejected under 35 USC 101 because the claimed invention is directed to non-statutory subject matter. Page 3 of the instant office action provides the following language:

> Claims 1-11 are non-statutory because they are directed to a system that does not have any physical structural elements..."A system having a processor is recommended."

Applicant has amended claim 1 to recite a system having a processor portion and a memory portion. Both are standard elements of a computer. Support for these structural elements exists throughout the specification, which recites a computer and standard computer operating systems throughout. For example paragraph [0087] references a computer and exemplary operating system and is reproduced here for convenience:

> [0087] The core data from which an embodiment of the present invention functions may include data and events obtainable on a computer running the Microsoft Windows operating system in any of its incarnations including but not limited to Windows, Windows 95, Windows NT, Windows XP, hereinafter referred to as Windows.

Applicant respectfully submits that claim 1, as amended, overcomes the present rejection and that dependent claims 2 though 11 accordingly also overcome the present rejection. Applicant therefore requests withdrawal of the present rejection.

Claims Objections

Claims 1 and 7 are objected to because of several informalities. Claims 2-6 and 8-11 are objected to for dependency on parent claim 1. To address informalities, Applicant has amended claim 1 to recite "mimics" instead of "mimic."

Additionally, Claim 7 as amended obviates the need for inserting "analysis" and "redirects." Amended claim 7 now contains the following amendments:

> 7. (Currently Amended) The system of ~~Claim 1~~Claim 6 further comprising executable steps following the filtering step~~:~~
> ~~a filtering means wherein said filtering means analysis~~ of analyzing the API calls and associated parameters and Internet Protocol network event data passed to and from the API calls~~ the trace performed,~~ and producing an XML extract file comprising ~~said analysis produces~~:
> an XML record for each content object in ~~the~~ temporal order of receipt ~~the content was received~~;
> an XML record redirect ~~records~~ record and added redirect information;
> an XML record for cookie reads;
> an XML record for cookie writes;
> an XML record for user navigation events;
> an XML record for HTTP header information~~ as XML~~; and
> ~~various other~~ one or more management information records relating to the API calls and associated parameters and Internet Protocol network event data passed to and from the API calls~~ to the network and user event traces~~.

Claim 7 as amended contains no new matter. Support for these amendments exists at least at paragraphs [0027] and [0089] of the Specification. Paragraph [0027] describes tracing API calls and related data events on the IP network layer and is reproduced above in regard to amendments to the Specification. Paragraph [0089] of the Specification provides support for recording content in temporal order, as claimed in amended claim 7:

> [0089] In addition to the original raw trace data, an embodiment of the present invention may insert markers or tags in the trace that provide added information about events and data. The markers or tags may include ***timing***, and other information used by an embodiment in the analysis and other phases. [Emphasis Added]

Applicant respectfully submits that claim 7 as amended therefore is in condition for allowance, and Applicant respectfully requests withdrawal of the present objection.

Rejections under 35 USC 112, second paragraph

Claims 1 through 20 are rejected under 35 USC 112, second paragraph as being indefinite for failing to particularly point out and distinctly claim the subject matter which applicant regards as the invention. Pages 4 and 5 of the instant office action recite various items requiring clarification throughout the claims. Applicant accordingly has revised the claims. The Specification provides support for all amendments and, as described below in detail, none of the amendments to the claims constitute new matter.

First, **amended claim 1** now includes structural indentations and layered enumerations that improve readability.

Next, Applicant has amended claim 1 to recite "website" instead of "site" which corresponds with the Examiner's interpretation. Support for this term exists through the Specification, for example at paragraphs [0023] and [0027] which are reproduced above in the **Amendments to the Specification** section of this response.

Claim 1 as amended includes amendments to the terms "tracing means" and "filtering means" such that the executable steps of the computer software program comprise steps of "tracing" and "filtering." Claim 1 as amended also includes a reference to "at least one existing website," which replaces the former limitation of "the site."

Amended claim 1 also now recites structural computing elements that clarify how the source code and robot are generated and how such source code, robot, computer software program, tracing means, and filtering means are related. As mentioned in response to the rejections under 35 USC 101, support for the structural amendments added to claim 1 exists at least at paragraph [0087]. Furthermore, support for the automatic generation of the software robot source code exists throughout the specification. For example, paragraphs [0093] and [0094] of the Specification recite the following language:

> [0093] Again referring to FIG. 1, once the trace 3 is filtered 4, an analysis 5 of the filtered trace data may be preformed. The output of the initial analysis is XML 9. The XML 9 is referred to as the "Extract"...[0094]...Generations of source code

> 31 may be via an XSL transform of the Extract XML. Another embodiment for generating source code or other useful expressions of the extract may be by modification of the source code 33 that may be performed in other ways including hard coding or using a computer language to parse the extract and generate some other useful product from the extract.

Furthermore, paragraph [0029] succinctly describes the present invention and automatic generation of source code:

> [0029] In an embodiment, the method further comprises the step of automating the generation of source code to an end user.

Applicant also respectfully submits that the entire Specification is directed toward a software application program that automatically generates a source code through the steps of tracing and analyzing API calls and corresponding Internet Protocol network event data. For example, see paragraph [0091], reproduced above with regard to **Amendments to the Specification**.

Applicant respectfully submits that element b) of claim 1 as currently amended also finds support in the specification.

> b) the automatically generated executable software robot stored in the memory portion for execution by the processor portion when an end user requests playback.

First, paragraph [0008] of the specification defines a "software robot."

> [0008] Programs that access web sites, that are normally accessed by human users sitting at a computer using a web browser, are call variously `software robots`, `robots`, `software agents`, `agents`, `programs` or other. Programs that access web sites appear from a network and data presentation format, to the web site as an end user would.

Next, as stated in Specification paragraph [0029], reproduced above, the current invention automates the generation of source code to an end user. Logically, the source code provided to the end user is the software robot. Furthermore, support exists for the element of end user "playback" throughout the Specification which describes automatic generation of a software robot for use by an end user. For example, paragraphs [0055] and [0099] explicitly describe a method of playback:

> [0055] An advantage of the present invention is to provide an automated software robot generator and a method for using the same wherein the software may modify a source code to include a playback function.
> [0099] FIG. 3 illustrates the relationship of various software modules of the computer software program 1 in an embodiment. The software modules may be divided into those that run during the analysis data capture phase, and software that may run either on a client and/or on a server ***machine during playback of generated source code 31***. [Emphasis Added]

Element b) of claim 1 as amended therefore constitutes no new matter. Addition of this element is for the purpose of better claiming the interoperability of the system elements. Applicant respectfully submits that amended claim 1 also now recites structural computing elements that clarify how the source code and robot are generated and how such source code, robot, computer software program, tracing means, and filtering means are related.

Amended claim 2 is cancelled, thereby obviating the need to revise that claim for clarity. The subject matter of claim 2 is incorporated into independent claim 1 as currently amended.

Amended claim 3 no longer recites the term “tracing means” and instead provides further limitation of claim 1 with regard to the Internet Protocol network event data. Support for the amendment to claim 3 exists at least at paragraphs [0088] and [0091] of the Specification, reproduced above with regard to **Amendments to the Specification**.

Amended claim 4 now contains language consistent with independent claim 1 so that the form term “computer program” now reads “computer software program.”

Amended claim 5 no longer contains a reference to the computer software program, thereby obviating the need to clarify the deleted language.

Amended claim 6 corresponds with amended claim 1 and now recites “the executable step of filtering” instead of “filtering means.” Amended claim 6 no longer includes the limitation “said information” and the term “HTTP messages” is amended to correspond with the earlier introduced term “HTTP based messages.” The term “current object” now reads “content object” in accordance with the Specification at, for example, paragraph [0092], reproduced above with regard to **Amendments to the Specification**.

Amended claim 7 no longer includes a reference to "filtering means." The limitation "the content" is also cancelled from the claim. Furthermore, claim 7 now depends from amended claim 6 so that the term "each content object" has antecedent basis.

Amended claim 8 now depends from claim 7 and adds further limitation to the XML extract file introduced in claim 7. Amended claim 1, from which claim 8 indirectly depends, recites an executable step of "generating a source code." Amended claim 8 no longer contains the terms "the computer program" and "during tracing." Support for amendments to claim 8 exists at least at Specification paragraphs [0093] and [0094].

Amended claim 9 now depends from claim 7 and adds further limitation to the XML extract file introduced in claim 7. Amended claim 1, from which claim 9 indirectly depends, recites an executable step of "generating a source code." Amended claim 9 no longer contains the terms "the computer program" and "during tracing." Support for amendments to claim 9 exists at least at Specification paragraphs [0093] and [0094].

Amended claim 10 now depends from claim 8 and adds further limitation to the XSL transform of the XML extract file introduced in claim 8. Support for amendments to claim 10 exists at least at Specification paragraphs [0093] through [0095].

Amended claim 11 now corresponds with amended claim 1 and includes the term "the at least one existing website."

Amended claim 12 no longer recites the terms "existing websites" and "traced." Additionally, claim 12 as amended now recites language that explicates how the source code is generated and how generating source code is related within the computer software program tracing, filtering and analyzing steps. Amendments to claim 12 add no new matter.

First, claim 12 now includes structural indentations and layered enumerations that improve readability.

Next, the preamble and item a) of amended claim 12 include references to "a computer network" and to a system in communication with the computer network and comprising standard computing elements. Support for these amendments exists at least at paragraph [0085] of the Specification, reproduced here in pertinent part for convenience (including amendments):

> [0085]...In Figure 1, the program 1 may capture Internet protocol and application data flowing over the network between the application being traced and one or more computers on a network attached directly or indirectly to the computer running the ~~AFT~~ computer software program.

With regard to amendments to executable steps i) through iii), support for these amendments exists throughout the specification. For example, paragraph [0088] recites language related to tracing API calls and associated parameters and data. Paragraph [0092] recites language with regard to filtering the data, and paragraphs [0093] and [0094] recite language with regard to analyzing the data and producing an extract file for generating source code. (For convenience, all of these paragraphs are reproduced in this response, at least in pertinent part, in sections entitled **Amendments to the Specification** and **Amendments to the Drawings**.) Additionally, paragraph [0062] describes that the data is in the form of packets in "temporal order." (One of ordinary skill in the art would understand data associated with API calls over the IP layer would transfer as IP data packets and that the term "packets" refers to IP data packets.)

Enumerated items iv) and b) recite automatically generating a software robot that mimics a system user's interactions with the existing website.

> [0029] In an embodiment, the method further comprises the step of automating the generation of source code to an end user.

Support for this language exists at least at claim 1 and Specification paragraphs [0029] and [0094] (reproduced above with regard to **Amendments to the Drawings** and with regard to rejection of claim 1 under **35 USC 112, Second Paragraph)**. Additionally, Specification paragraph [0008] defines the term "software robot" and Specification paragraphs [0010] through [0012] define the problem solved by the software robot of the present invention:

> [0008] Programs that access web sites, that are normally accessed by human users sitting at a computer using a web browser, are call variously `software robots`, `robots`, `software agents`, `agents`, `programs` or other. Programs that access web sites appear from a network and data presentation format, to the web site as an end user would. The web site or web application receives and sends information exactly as it would to a real human user, but instead the information is received, processed, and responded to by a piece of software running on a computer somewhere on the Internet, rather than a live human user using a web browser. In these applications, the web site, or web application does not know it is communicating with anything other than a web browser, when in fact it is communicating with a piece of custom written software...
>
> [0010] No invention exists today that automatically generates software robots that can manipulate existing web sites and web applications. The net effect of the invention is a dramatic reduction in the amount of time necessary to implement machine to machine communications that utilize existing web sites and web applications.
>
> [0011] Moreover, no application or program exists today that uses a system of monitoring and analyzing functions placed to a distinct web browser or other network based application to produce an extract of network transactions that can be manipulated by software to perform the desired operation automatically.
>
> [0012] Further, no system of application exists for a program to emulate the transactions of the network based application and mimic the transactions in later access of the same network based applications or other relevant network abilities.

Additionally, paragraph [0034] of the Specification clearly states that the present invention provides, "...an automated software robot generator and a method for using the same." Inclusion of the term "software robot" in amended claim 12 for the purpose of further defining the automatic generation of source code is consistent with the provided definition and adds no new matter. Introduction of automatically instructing the system to mimic network transactions and manipulate websites and web applications also finds support at least in these referenced sections of the Specification.

Element b) of claim 12 as currently amended also recites that the software robot is "adapted for playback on the IP network level at the request of an end user." Applicant respectfully submits that this element adds no new matter. As stated with regard to claim 1 as currently amended, the source code provided to the end user is the software robot with source code generation based on IP network level events. Furthermore, support exists for the element of end user "playback" throughout the Specification which describes automatic generation of a software robot for use by an end user. For example, paragraph [0055] explicitly describes a method of playback:

> [0055] An advantage of the present invention is to provide an automated software robot generator and a method for using the same wherein the software may modify a source code to include a playback function.

Element b) of claim 12 as amended therefore constitutes no new matter. Addition of this element is for the purpose of better claiming the interoperability of the elements. Applicant respectfully submits that amended claim 12 also now recites structural computing elements that clarify how the source code and robot are generated and how such source code, robot, computer software program, tracing means, and filtering means are related. Applicant respectfully submits that the amendments to independent claim 12 find support throughout the Specification and therefore add no new matter.

Amended claim 13 no longer includes the limitation "existing websites" and now recites recordation of IP network data passed to and from API calls. Support for the amendment to claim 13 exists at least in paragraph [0088] of the Specification, reproduced here in pertinent part:

> [0088] A software hook may work by intercepting Application Programming interface calls, analyzing and possibly recording the parameters and data passed to the API call as shown in FIG. 2...the software may work by obtaining a detailed trace of various API calls and the parameters and data sent to and from those API's. The trace data may be saved in memory (RAM) or on a disk (see FIG. 2).

Applicant respectfully submits that amended claim 13 contains no new matter.

Claim 14 is cancelled. The subject matter of claim 14 is incorporated into independent claim 12 as currently amended.

Amended claim 15 and **amended claim 16** contain no new matter and describe the transformation of the XML extract data. Support for these amendments exists at least at paragraph [0094] of the Specification, reproduced here in pertinent part:

> [0094] Generations of source code 31 may be via an XSL transform of the Extract XML. Another embodiment for generating source code or other useful expressions of the extract may be by modification of the source code 33 that may be performed in other ways including hard coding or using a computer language to parse the extract and generate some other useful product from the extract.

Amended claim 17 no longer recites the terms "computer program" and "underlying site." Support for amendments to claim 17 exists at least at paragraphs [0030] and [0097] of the Specification, which reference interfacing with a website and automatically generating the listed actions if filling in forms, obtaining information, performing system testing and monitoring websites for change.

Amended claim 18 no longer recites the limitation "software." Instead, claim 18 recites "executable steps" in accordance with amended claim 12. One skilled in the art would know that software comprises a series of executable steps, or instructions. Amended claim 18 also includes a revision of the term "web" to "website." These amendments add no new matter.

Amended claim 19 no longer recites the limitation "software" and instead recites "executable steps" in accordance with claim 12. Amended claim 19 now depends from claim 18 and further limits the "plug-in" introduced in claim 18. Support for amendments to claim 19 exists at least at paragraph [0096], reproduced above with regard to **Amendments to the Specification**. Additionally, the term "debug messages" is further defined by the language "adapted for providing full interactive debugging capability to the source code." Support for this definition exists at least at paragraphs [0076] and [0077] of the Specification:

> [0076] Still another advantage of the present invention is to provide an automated software robot generator and a method for using the same wherein a web browser plugin is used ***to provide full interactive debugging capability to the software code that is generated during the tracing and code generation phases***. [Emphasis added]

> [0077] Yet another advantage of the present invention is to provide an automated software robot generator and a method for using the same wherein a web browser plug provides the ability to do most common debugging functions ***on the generated source code***, while viewing the affect of the source code statements in the web browser itself. [Emphasis added]

Inclusion of this definition therefore adds no new matter.

Amended claim 20 is amended for proper method claim format. Additionally, the amended language finds support in the specification at least at paragraphs [0033] and [0097], which describe a plug-in button and form completion without further user intervention.

Rejection under 35 USC 102(e)

Claims 12-16, 18 and 20 have been rejected under 35 USC 102(e) as being anticipated by US Publication No. 2005/0114757 to Sahota et al. ("Sahota et al.") (Because claims 13-16, 18 and 20 depend upon independent claims 12, they include all limitations of that dependent claim. This response, therefore, addresses the present rejection with regard to independent claim 12.) More specifically, language on pages 6 and 7 of the instant office action states that Sahota et al. anticipates every element of independent claim 12:

> Per claim 12: Sahota discloses:
> -manipulating an existing website, providing a computer software program to collect data on existing websites (i.e. "acquiring and transforming existing...HTML content...for display and execution," abstract)
> -tracing an existing website when the website is being accessed by third party users (i.e. "The content acquisition subsystem of the agent spider is flexible and new acquisition modules can be easily plugged in...to locate, acquire and convert content dynamically," page 5, 0059)
> -filtering the collected data on existing websites (i.e. page 9, 0103; page 5, 0058);
> -analyzing the data collected from traced and filtered websites (i.e. page 3, 0041, 0042)
> -generating a source code. (i.e. "an XML file is transformed into an HTML web page," page 6, 0068 and 0069).

Applicant respectfully traverses this rejection. Sahota et al. fails to teach every element of the computer implemented method of Applicant's independent claim 12, as currently amended. In particular, Sahota et al. fails to teach tracing API calls in temporal order between the system and the existing website, and tracing associated parameters and data associated with Internet Protocol (IP) network events passed to and from the API calls when a system user accesses the website. Furthermore, Sahota et al. fails to teach generating a software robot that comprises executable source code for automatically mimicking interactions between a user and a website. Independent claim 12 as currently amended now recites these features at least in subparagraphs i) and b). These amendments add no new matter and support exists at least at Specification paragraphs [0094], [0008], [0010] through [0012], [0034], and [0062] as discussed above in response to rejections under 35 USC 112, second paragraph.

Instead of tracing IP network event data and associated parameters passed to and from Application Programming Interfaces (APIs) making calls while a system user accesses a website, the invention of Sahota et al. focuses on using web browsers, and in particular a web-page markup language, HTML, to generate web page layout on various platforms. For example, the abstract of Sahota et al. and paragraph [0059] describe this browser-level invention in the first sentence:

> A method and system are disclosed for acquiring and transforming existing content (e.g., Hyper Text Markup Language HTML content) for display and execution on multiple platforms and architectures....[0059] Agent spider 207 uses existing standard HTML parser engine 217 to read and transform the structure and content of any given page.

Language at page 6, paragraph [0068] further describes this HTML transformation by the Sahota et al. invention:

> [0068] At operation 206, the XML file or document can be transformed into a displayable format. For example, content converter 204 and content generator 203 can be used together to transform an XML file stored in XML data files 208a. In one embodiment, an XML file is transformed into an HTML web page.

Unlike the present invention, Sahota et al. fails to teach generation of a software robot that comprises executable source code that mimics user interactions with an existing website. Instead, Sahota et al. teaches transforming an XML file into markup language, which is a display format language, not an executable source code. Merriam Webster Online Dictionary provides a concise definition of "markup language":

> : a system (as HTML or SGML) for marking or tagging a document that indicates its logical structure (as paragraphs) and gives instructions for its layout on the page especially for electronic transmission and display

In contrast, Merriam Webster Online Dictionary provides the following definition of source code:

> : a computer program in its original programming language (as FORTRAN or C) before translation into object code usually by a compiler

Sahota et al. captures "data" associated with the markup language rather than IP network event data. For example, page 3, paragraph [0041] of the Sahota et al. reference defines this HTML-related data:

> At operation 154, data from the content is extracted using the created capture templates. For example, content harvest and conversion platform 130 can be used to extract pure data can be extracted from the web page such as, for example, ***the HTML tags and attributes***. [Emphasis added.]

The invention of Sahota et al. thus focuses on web page layout and markup language detailing that layout, instead of tracing IP data packets to and from API calls on the Internet Protocol network level of system architecture (for example, the TCP/IP and OSI models) to produce a software robot comprising executable source code.

In contrast, the present invention traces a temporal sequence of IP network layer data packets transferred to and from API calls while a user is accessing an existing website from a computer processing executable steps. This IP network layer trace is a critical component with which the present invention produces source code that mimics user interactions with an existing website. Sahota et al. fails to teach at least this critical limitation of independent claim 12 as currently amended. Applicant therefore respectfully submits that independent claim 12 is in condition for allowance. Because claims 13-16, 18, and 20 depend from claim 12 and include all limitations of that independent claim, Applicant respectfully submits that these dependent claims are also in condition for allowance. Applicant requests reconsideration and withdrawal of the present rejection.

Rejections under 35 USC 103(a)

Claims 1-11 have been rejected under 35 USC 103(a) as being unpatentable over US Publication No. 2005/0114757 to Sahota et al. ("Sahota et al."). Pages 9 and 10 of the instant Office Action describe Sahota et al. as teaching every element of independent claim 1 except for the generation of a software robot. With regard to that element, the Office Action provides the following language:

> Sahota does not explicitly teach that the generated code is a software robot that mimics a user using a web browser on the site. However, XSL that transforms the XML extract into other source formats would allow the XML file to be transformed in any other code (i.e. page 6, 0068 and 0069). Therefore, it would have been obvious for one having ordinary skill in the art to modify Sahota's disclosed system at the time applicant's invention was made to generate agent spider code from the XML extra by XSL for manipulation of existing websites on behalf of users.

Applicant respectfully traverses this rejection. Sahota et al. fails to teach or suggest the limitations of independent claim 1 as currently amended.

As described above with regard to the rejection of independent claim 12 under 35 USC 102(e), instead of tracing IP network event data and associated parameters passed to and from Application Programming Interfaces (APIs) making calls while a system user accesses a website, the invention of Sahota et al. focuses on using web browsers, and in particular a web-page markup language, HTML, to generate web page layout on various platforms. Unlike the present invention, Sahota et al. fails to teach generation of a software robot that comprises executable source code that mimics user interactions with an existing website.

Instead, Sahota et al. teaches transforming one markup language, which is a display format language distinct from an executable source code, to another markup language (i.e., transforming Extensible Markup Language (XML) into Hypertext Markup Language (HTML)). Sahota et al. does not teach, motivate or suggest the transformation of a markup language to executable source code. Further, one skilled in the art of software robots comprising automatically generated executable source code would not look to the web browser invention of

Sahota et al. to teach, motivate or suggest tracing IP data packets on the network level to generate a software robot that mimics user interactions with a website.

Applicant respectfully submits that because Sahota et al. fails to teach or suggest the limitations of Applicant's independent claim 1 as currently amended, independent claim 1 is in condition for allowance. Because claims 2 through 11 depend from independent claim 1 and include all limitations thereof, applicant respectfully submits that those dependent claims are also in condition for allowance. Applicant respectfully requests reconsideration and withdrawal of the present rejection.

Claims 17 and 19 have been rejected under 35 USC 103(a) as being unpatentable over Sahota et al. in view of US Patent No. 7,231,606 to Miller et al. ("Miller et al."). Because these claims depend from independent claim 12, this response will address the present rejection on the independent claim level. As described above with regard to the rejection under 35 USC 102(e), Sahota et al. fails to teach and suggest independent claim 12 as currently amended. Namely, Sahota et al. fails to provide any motivation for teaching the automatic generation of source code based on tracing, filtering and analyzing IP network event data to produce a software robot that mimics user interactions with a website.

Miller et al. fails to cure the deficiency of Sahota et al. and the proposed combination of references fails to teach or suggest independent claim 12 as currently amended. Specifically, independent claim 12 teaches the automatic generation of a software robot comprising source code that mimics interactions between the system user and the existing website wherein the software robot is adapted for playback on the IP network level at the request of an end user. In contrast, Miller et al. teaches creating a test script for testing a website via a test-enabled Internet browser for ensuring repeatable results of the website under test. Column 2, lines 59-57, reproduced here in pertinent part, describe the invention of Miller et al.:

> According to a general aspect of the invention website testing is performed in a browser environment....The invention thus can be implemented in a test-enabled Internet browser.

Miller et al. teaches a user-directed recordation of webpage results to create a user-defined script for testing a webpage at the test-enabled web-browser level. For example, column 3, lines 21-25 and lines 56-59 describe this high level of user involvement in the recordation of the test script:

> Here, a test-enabled web browser is used in the normal way to interact with the website under test. During this interaction, the user can command certain kinds of data to be extracted into a script file as a part of the recording process...During recording a user can select and specify certain information that is extracted immediately from the current rendition of a web page and which is deposited in special formats within the recorded script.

Not only does Miller et al. focus solely on the web browser level, but the reference fails to teach the steps of tracing, filtering and analyzing IP network data and parameters passed to and from API calls to automatically generate source code comprising a software robot. Moreover, like Sahota et al., Miller et al. fails to teach automatically generating any software robot. Miller et al. teaches away from automatically generating a source code by tracing, filtering and analyzing IP network event data and instead teaches creation of a script recorded through user commands during interaction with a test-enabled web browser. Those user commands are vital to practicing the invention of Miller et al. and providing a means for validating the website under test.

Further in support of this distinction is that the invention of Miller et al. is limited in scope to recording user-selected events in the Internet Explorer test browser, or to recording windows operating system function key presses on a keyboard and mouse clicks. Miller et al. fails to teach, suggest or provide any motivation for creating executable steps enabling network tracing for automatically generating a software robot. For example, language at column 9, lines 7 through 23 and lines describe the focus of Millers et al. as centering entirely on a test-enabled web browser:

> 13. Record/Play--Application Mode
> During the recording process the user can signal a change in internal recording state using the control GUI from the normal recording mode to Application Mode recording. This mode has the advantage that it can apply not only to activity within the browser window but also to any other application launched by the test enabled web browser that exists on the Windows desk-top. In application mode recording only keyboard activity, mouse click, and mouse drag activity is recorded. In application mode the only validation modes are a partial image synchronization and validation of text that is selected by the user and put into the clipboard, a logical area that is maintained by the underlying Windows environment, that contains the results of a COPY instruction.

Here, Miller et al. describes a "normal mode" which performs browser based recording, and "application mode" in which the invention records only keystrokes and mouse events. No suggestion is made of recording network level tracing or how to proceed with such IP level network tracing.

Language at column 9, lines 40 through 67, reproduced here in pertinent part, further describe the teaching of Miller et al. as focusing solely on WINDOWS screen events:

> 15. Record/Play--Multiple-Window (Sub-Browser) Operation
> In accordance with the representative environment the special web browser must test WebSites that evoke multiple windows reliably... In the representative environment this is accomplished by automatically opening a second instance of the testing browser, one already designated to be in record mode, whenever the parent page requests a new window. In the representative environment resulting sub-script is recorded in an script file named automatically and systematically by eValid according to the sequence number of the sub-browser that was launched and the depth of nesting of the recording sub-browser... During playback the representative environment automatically launches a copy of the special browser in response to the LaunchSubBrowser command and instructs the launched sub-browser to play back the indicated sub-script. This accomplishes the effect of simultaneous parent and child playbacks exactly as was recorded initially.

Here, Miller et al. clearly fails to teach network based tracing/recording with regard to the recording of interactions of "sub browsers". Miller et al. teaches away from network based tracing by explicitly launching sub browsers during playback of the script rather than by directly generating IP network calls to perform the playback.

Miller et al. clearly teaches away from tracing IP network events and automatically generating a source code adapted for playback at the network level upon request by an end user. In contrast, the present invention traces events directly at the network layer and therefore enables playback in the absence of a browser or other application to simulate user interactions with a website. Running in a web browser as a plug in is only one option for playback of the automatically generated source code of the present invention and is not the only option. For example, at least FIG. 1 and Specification paragraph [0099], reproduced here in pertinent part, describe the ability of the software robot of the present invention to run either as a web browser plug in OR to run as an independent source code operating independent of a web browser:

> [0099] ~~Figure~~FIG. 3 illustrates the relationship of various software modules of the computer software program 1 in an embodiment. The software modules may be divided into those that run during the analysis data capture phase, and software that may run either on a client and/or on a server machine during playback of generated source code 31. ***Client playback may be either via a dedicated win32 application 34, or a web browser plug-in 35 and/or browser helper object.*** [Emphasis added]

Furthermore, with regard to claim 19, Miller et al. fails to teach the debug function of the present invention. Claim 19, as currently amended, recites the debug function as adapted for debugging the automatically generated source code. Miller et al. fails to teach or suggest generating source code. The "debug" functions described at column 5, lines 48-67 involve error messages related to testing a website for repeatable results:

> The test-enabled browser can perform a variety of different tests for websites. One such test is a load test, in which loads are artificially supplied to a particular website to test its performance and abilities under such circumstances...The test-enabled browser 302 can produce various reports and can log various different items. For example, as shown in FIG. 3, the test-enabled browser 302 can produce an event log, a performance log, a message log, a timing log, and a load test log. The various logs can also be converted into more user-friendly charts. The test-enabled browser 302 can also produce reports, such as a site map report. Still further, the test-enabled browser 302 can record, edit and load scripts.

The "logs" produced by Miller et al. thus relate to load testing a website. These are non-analogous to the debugging function of the present invention which allows for debugging of the automatically generated source code.

Because Miller et al. fails to cure the deficiencies of Sahota et al., the proposed combination of references fails to teach or suggest every element of independent claim 12 as currently amended. Applicant respectfully submits that independent claim 12 as currently amended is in condition for allowance and that dependent claims 17 and 19, which depend from claim 12 and include all limitations of that independent claim, are also in condition for allowance. Applicant respectfully requests reconsideration and withdrawal of the present rejection.

Listing of Claims Without Showing Markings

1. (Currently Amended) A system in communication with a computer network for manipulating at least one existing website displayable over the computer network, the system comprising:

a) a processor portion and a memory portion having a computer software program stored thereon that comprises steps executable by the processor portion, wherein the executable steps comprise:

i) accessing the at least one existing website as directed by a user of the system;

ii) tracing API calls by intercepting associated parameters and Internet Protocol network event data obtained from one or more application programming interfaces while accessing the at least one existing website;

iii) filtering the Internet Protocol network event data;

iv) automatically generating a source code from the traced and filtered Internet Protocol network event data that is executable by the processor portion, and thereby automatically generating an executable software robot that mimics the user using a web browser to access the at least one existing website; and

b) the automatically generated executable software robot stored in the memory portion for execution by the processor portion when an end user requests playback.

2. (Cancelled)

3. (Currently Amended) The system of Claim 1 wherein the Internet Protocol network event data is obtainable from a group of APIs consisting of Winsock API, MICROSOFT WinInet API, MICROSOFT shell API, MICROSOFT security API, MICROSOFT User API, MICROSOFT Active Directory API, MICROSOFT HTML API, MICROSOFT DOM API and/or any combination thereof.

4. (Currently Amended) The system of Claim 1 wherein the computer software program traces Internet Protocol network event data obtainable on a computer running a MICROSOFT WINDOWS operating system.

5. (Currently Amended) The system of Claim 1 wherein the Internet Protocol network event data comprises data passed to and from a website browser application and/or a non-website browser application.

6. (Currently Amended) The system of Claim 1 wherein the executable step of filtering is adapted for removal of redundant and useless Internet Protocol network event data passed to and from the API calls and wherein the executable step of filtering further comprises:

removing network management packets that are acknowledgements and retries;

collating IP packets into single HTTP based messages; and

collating HTTP based messages into single records of content objects, wherein the content objects comprise HTML, images, audio, and other HTTP content.

7. (Currently Amended) The system of Claim 6 further comprising the executable steps of

analyzing the API calls and associated parameters and Internet Protocol network event data passed to and from the API calls, and

producing an XML extract file comprising:

an XML record for each content object in temporal order of receipt;

an XML redirect record and added redirect information;

an XML record for cookie reads;

an XML record for cookie writes;

an XML record for user navigation events;

an XML record for HTTP header information; and

one or more management information records relating to the API calls and associated parameters and Internet Protocol network event data passed to and from the API calls.

8. (Currently Amended) The system of Claim 7 wherein the executable step of generating source code further comprises transforming the XML extract file into executable source code via XSL.

9. (Currently Amended) The system of Claim 7 wherein the executable step of generating source code further comprises using a computer language to parse the XML extract file.

10. (Currently Amended) The system of Claim 8 wherein XSL transforms the XML extract file into source code written in a programming language selected from a group consisting of JAVA, JAVASCRIPT, VISUAL BASIC, COLD FUSION, C/C++, PASCAL and a plurality of other computer languages.

11. (Currently Amended) The system of Claim 1 wherein the software robot is adapted to interface with the at least one existing website and automatically manipulate the at least one existing website during use.

12. (Currently Amended) A method for manipulating an existing website in communication with a computer network, the method comprising the steps of:

a) providing a system, also in communication with the computer network, comprising a processor portion and a memory portion having executable steps stored thereon for execution by the processor portion, wherein said executable steps comprise:

i) tracing API calls in temporal order between the system and the existing website and associated parameters and data associated with Internet Protocol (IP) network events passed to and from the API calls when a system user accesses the existing website;

ii) filtering the data;

iii) analyzing the data to produce an extract file; and

iv) automatically generating a software robot that comprises executable source code derived from the extract file, wherein executing the source code parsed from the extract file automatically instructs the system to mimic interactions between the system user and the existing website; and

b) executing steps i) through iv), thereby automatically generating a software robot that manipulates the existing website by automatically instructing the system to mimic interactions between the system user and the existing website and that is adapted for playback on the IP network level at the request of an end user.

13. (Currently Amended) The method of Claim 12 further comprising a step following the filtering step of recording to the memory portion the API calls and associated parameters and data passed to and from the API calls.

14. (Cancelled)

15. (Currently Amended) The method of Claim 12 wherein the generating step comprises prompting an end user to hard code the source code from the XML extract file.

16. (Currently Amended) The method of Claim 12 wherein the generating step comprises automating the derivation of source code from the XML extract file.

17. (Currently Amended) The method of Claim 12 wherein the software robot is adapted for interfacing with the existing website to automatically fill-in one or more forms on a web page; obtain information previously unattainable; to perform system testing of the website; and/or to monitor the existing website for change.

18. (Currently Amended) The method of Claim 12 further comprising the step of integrating the executable steps into a website browser as a plug-in.

19. (Currently Amended) The method of Claim 18 wherein the source code is adapted for interactive stepping through on a page-by-page or event-by-event basis so that debug messages adapted for providing full interactive debugging capability to the source code, HTTP header parameters, and other data are displayed in the plug-in.

20. (Currently Amended) The method of Claim 12 further comprising a final step of displaying a selectable button in a web browser plug-in that automates completion of multiple forms without further user intervention.

Summary

In light of the above, Applicant respectfully requests consideration of the subject patent application. Any deficiency or overpayment should be charged or credited to Deposit Account No. 50-4514.

Respectfully submitted,

Kevin M. Farrell
Attorney for Applicants
Registration No. 35,505

April 3, 2008
Pierce Atwood, LLP
One New Hampshire Ave., Suite 350
Portsmouth, NH 03801
603-433-6300

ANNOTATED FIG 1

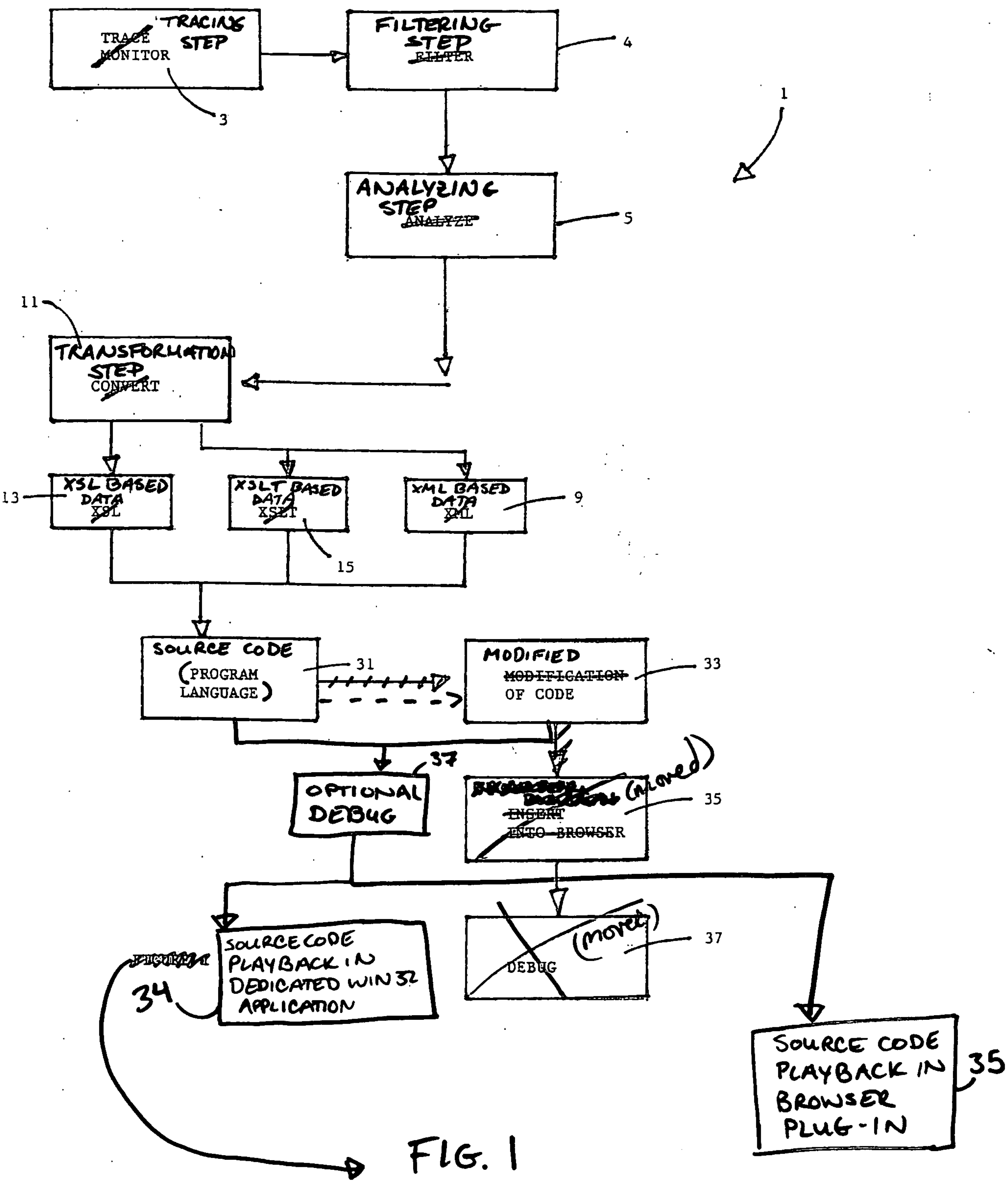
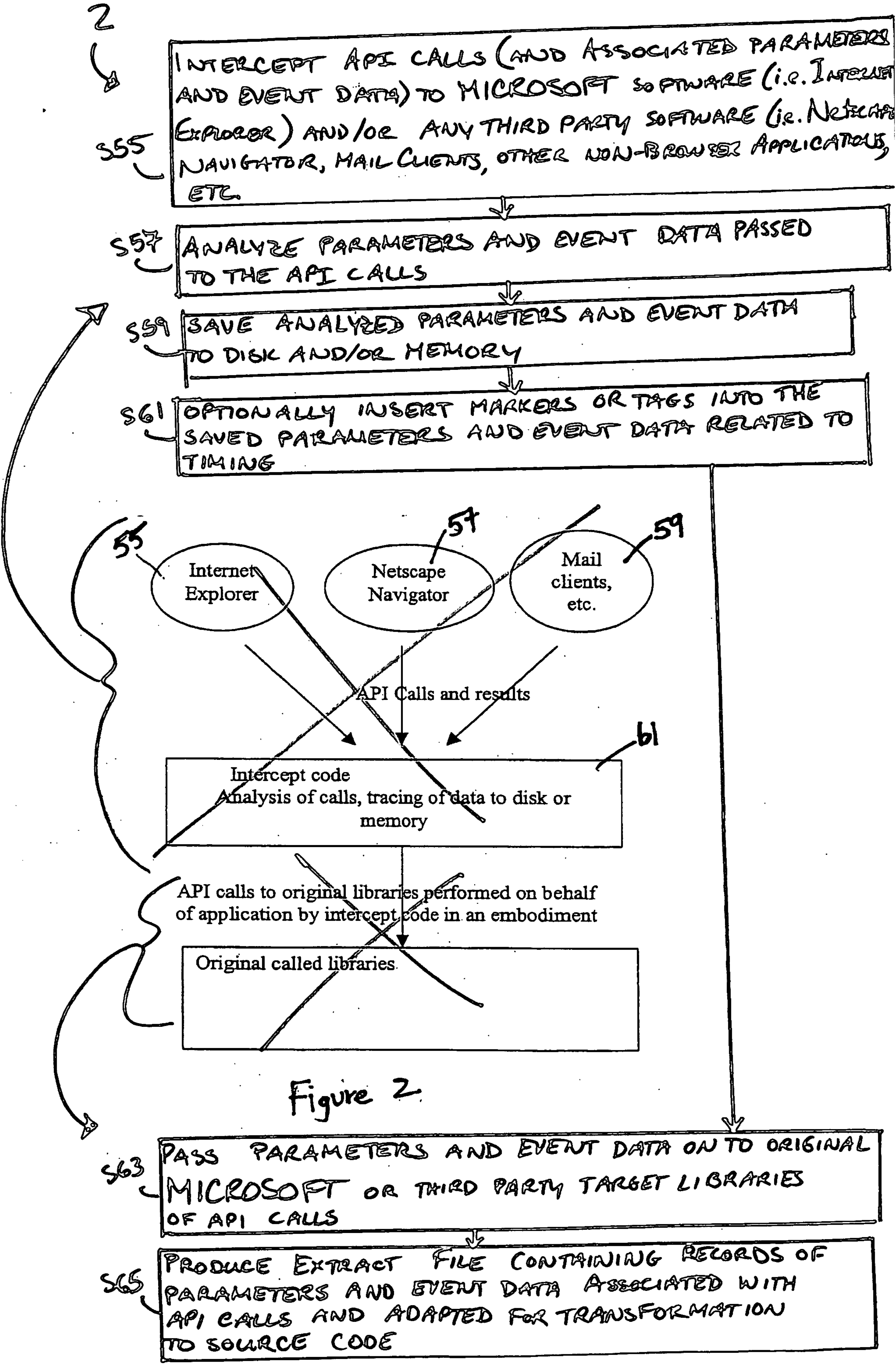


FIG. 1

ANNOTATED
FIG. 2



S55 INTERCEPT API CALLS (AND ASSOCIATED PARAMETERS AND EVENT DATA) TO MICROSOFT SOFTWARE (i.e. INTERNET EXPLORER) AND/OR ANY THIRD PARTY SOFTWARE (i.e. NETSCAPE NAVIGATOR, MAIL CLIENTS, OTHER NON-BROWSER APPLICATIONS, ETC.

S57 ANALYZE PARAMETERS AND EVENT DATA PASSED TO THE API CALLS

S59 SAVE ANALYZED PARAMETERS AND EVENT DATA TO DISK AND/OR MEMORY

S61 OPTIONALLY INSERT MARKERS OR TAGS INTO THE SAVED PARAMETERS AND EVENT DATA RELATED TO TIMING

S63 PASS PARAMETERS AND EVENT DATA ON TO ORIGINAL MICROSOFT OR THIRD PARTY TARGET LIBRARIES OF API CALLS

S65 PRODUCE EXTRACT FILE CONTAINING RECORDS OF PARAMETERS AND EVENT DATA ASSOCIATED WITH API CALLS AND ADAPTED FOR TRANSFORMATION TO SOURCE CODE

2
55
Internet Explorer
57
Netscape Navigator
59
Mail clients, etc.
API Calls and results
61
Intercept code
Analysis of calls, tracing of data to disk or memory
API calls to original libraries performed on behalf of application by intercept code in an embodiment
Original called libraries

Figure 2

ANNOTATED FIG. 3

Reproduced for clarity without changes

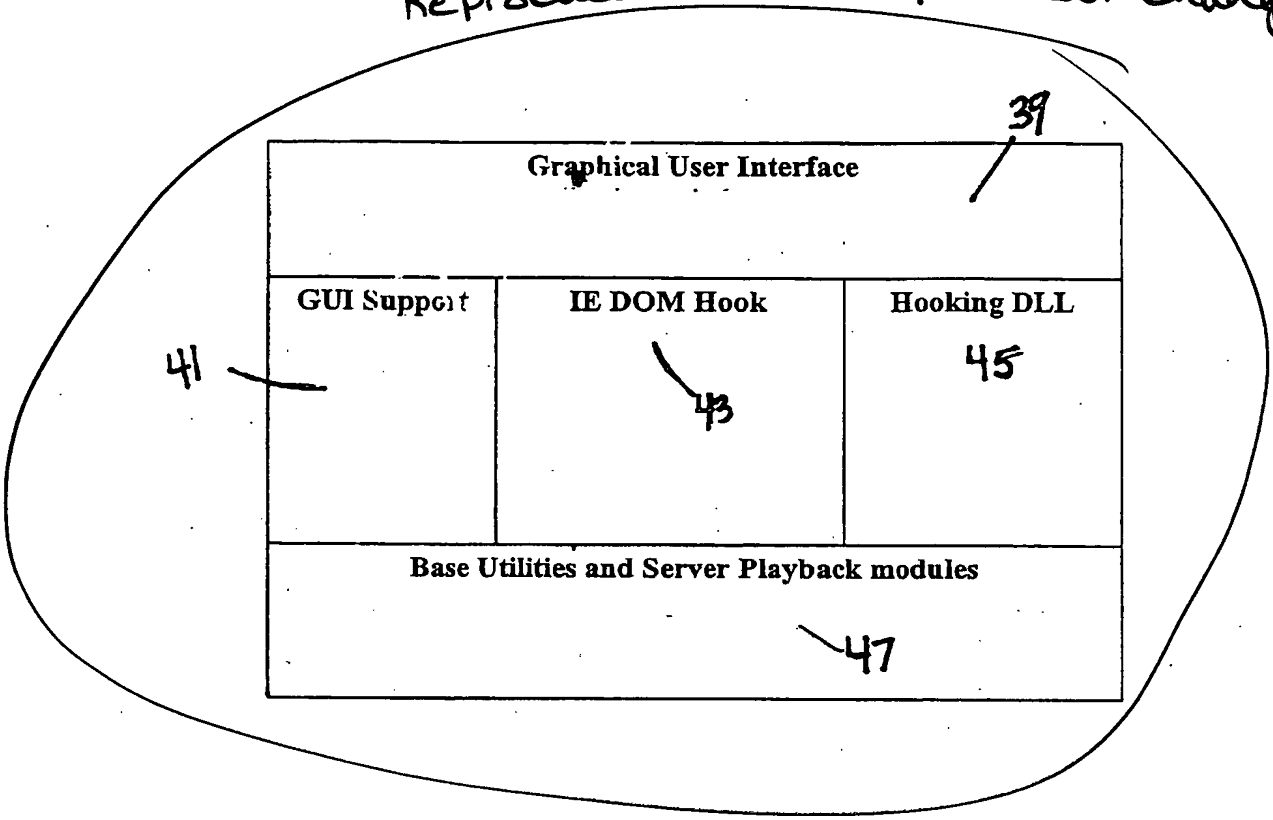


Figure 3